IN THE UNITED STATES DISTRICT COURT
FOR THE EASTERN DISTRICT OF MICHIGAN
SOUTHERN DIVISION

PROGME CORPORATION
208 Clair Hill Drive
Rochester Hills, MI 48309,

Plaintiff

v.

Civil Action No.

District Judge

JURY TRIAL DEMANDED

GOOGLE LLC
1600 Ampitheatre Parkway
Mountain View, CA 94043,

Defendant

COMPLAINT FOR PATENT INFRINGEMENT

Plaintiff Progme Corporation (hereinafter termed "Progme") files this **COMPLAINT FOR PATENT INFRINGEMENT** against Defendant Google LLC (hereinafter termed "Google") for infringement of U.S. Patent No. 8,713,425 ("425 Patent"). A copy of the '425 Patent is attached as **Exhibit A**.

THE PARTIES

1. Progme is a corporation existing under the laws of Michigan with its principal place of business at 208 Clair Hill Drive, Rochester Hills, MI 48309.
2. On information and belief, Defendant Google, a wholly owned subsidiary of Alphabet Inc., is a Limited Liability Company organized under the laws of the state of Delaware with its principal place of business at 1600 Ampitheatre Parkway, Mountain View, CA 94043. Defendant Google may be served in Delaware through its registered agent for service of process, Corporation Service Company, 2711 Centerville Road, Suite 400, Wilmington, DE 19808.

JURISDICTION AND VENUE

3. This action arises under the patent laws of the United States, 35 U.S.C. § 1, et seq. including 35 U.S.C. § 271. This Court has exclusive subject matter jurisdiction over this case for patent infringement under 28 U.S.C. §§ 1331 and 1338(a).

4. This Court has personal jurisdiction over Defendant Google for at least the following reasons: 1) Defendant Google has committed acts of patent infringement in Michigan and specifically in this Judicial District and 2) Defendant Google has purposefully established systematic and continuous contacts in this Judicial District and should reasonably expect to be haled into Court here.

5. Venue is proper in this district under 28 U.S.C. §§ 1391(b) and (c), and 1400(b) because Defendant Google regularly does or solicits business, engages in other persistent courses of conduct and/or derives substantial revenue from goods and services provided to individuals and/or businesses in Michigan and specifically in this Judicial District.

6. Venue is further proper in this Judicial District because, on information and belief, Defendant Google maintains offices and employees throughout the Detroit metropolitan area including a major office employing approximately 450 employees in Ann Arbor, Michigan and another major office employing approximately 100 employees in Birmingham, Michigan.

U.S. PATENT 8,713,425

7. On April 29, 2014, the U.S. Patent and Trademark Office duly and legally issued U.S. Patent No. 8,713,425 ("the '425 Patent"), entitled "AUDIO/VIDEO PROGRAM-RELATED HYPERLINK PRINTER", to Progme as assignee after a full and fair examination.

8. As indicated in the appended **Exhibit B**, Progme became the owner of all rights, title and interest in and to the ’425 Patent by recorded assignment and possesses all rights of recovery under the ’425 Patent, including the right to sue and recover damages for all infringements. Since the date of said assignment, Progme has been and remains the sole owner of said rights, title and interest in and to the ’425 Patent.

9. Progme operates the website PrintHD.TV listing various sets of one or more print() or println() statements of the PrintWriter method covered by the ‘425 Patent.

10. The ’425 Patent discloses and claims, in part, a method for generating and encoding and an apparatus for receiving and processing an hyperlink address string structured as a PrintWriter method having a pw, out, writer or ps parameter for hyperlinking to a resource in the initial array position of a list in which resource identifiers uniquely identifying resources corresponding to predetermined program material are arrayed printing predetermined printable output of said resource in the initial array position of a list in which resource identifiers uniquely identifying resources corresponding to predetermined program material are arrayed wherein said resource in the initial array position of a list in which resource identifiers uniquely identifying resources corresponding to predetermined program material are arrayed is defined within a print() or println() statement of a PrintWriter method. *See* ’425 Patent at 1:51-65 and 15:27-35 and claims 1, 2, 4 , 14 and 24.

11. As described in the ‘425 Patent, a predetermined hyperlink address comprises one or more resource identifiers in a list of resource identifiers in which resource identifiers uniquely identifying resources defined within print() or println() statements of the PrintWriter method wherein the resource in the initial array position of the list in which

resource identifiers uniquely identifying resources corresponding to predetermined program material are arrayed comprises a begin timing attribute. *See* '425 Patent at 1:51-65 and 15:27-35 and claims 1, 2, 4 , 14 and 24.

12. The '425 Patent further discloses and claims, in part, said hyperlink address string comprising a first attribute, comprising a begin timing attribute, indicating a predetermined hyperlink address comprising a resource identifier identifying said resource in the initial array position of a list in which resource identifiers uniquely identifying resources corresponding to predetermined program material are arrayed and a second attribute indicating one or more parameters defining predetermined printable output of said resource in the initial array position of a list in which resource identifiers uniquely identifying resources corresponding to predetermined program material are arrayed comprising at least one parameter instructing a PrintWriter to print said predetermined printable output of said resource in the initial array position of a list in which resource identifiers uniquely identifying resources corresponding to predetermined program material are arrayed for transmission in conjunction with program signals representative of predetermined program material wherein said resource identifier is used to hyperlink to a resource corresponding to said predetermined program material and said predetermined printable output of said resource in the initial array position of a list in which resource identifiers uniquely identifying resources corresponding to predetermined program material are arrayed is printed via said PrintWriter. *See* '425 Patent at 1:51-65 and 15:27-35 and claims 1, 2, 4, 9 and 14.

13. The '425 Patent further discloses and claims, in part, predetermined activation of said hyperlinking to said resource in the initial array position of a list in which resource

identifiers uniquely identifying resources corresponding to predetermined program material are arrayed combining activation of said hyperlinking with activation of said printing so that said hyperlinking prints predetermined printable output of said resource in the initial array position of a list in which resource identifiers uniquely identifying resources corresponding to predetermined program material are arrayed without said printing being separately activated after said hyperlinking is activated. *See* '425 Patent at 1:33-40, 1:51-65 and 15:27-35 and claims 1, 2, 4, 12, 14, 21 and 22.

14. The '425 Patent further discloses and claims, in part, said predetermined hyperlink address to predetermined hyperlinked content indicated in said first attribute of said hyperlink address string structured as a PrintWriter method comprising a resource identifier to identify i) a resource from resources of threads performed in a Java Virtual Machine (JVM) and ii) an application, in which certain thread objects belong or a resource consumer related to the resource belongs, executed in said JVM; a resource consumer for each thread using a resource in said JVM; a resource manager to manage resource usage wherein said resource identifier and resource manager are generated using Java language and registered resource managers are stored according to types and a resource allocation policy comprising calling a resource identifier that is in an initial array position of a list in which resource identifiers uniquely identifying resources are arrayed or requesting an initial resource identifier that is first on an array in a resource identifiers list. *See* '425 Patent at 21:6-67 and claims 2, 4 , 14 and 24.

15. After disclosing that said resource identifier is in said initial array position or said initial resource identifier is first on an array in a resource identifiers list, the '425 Patent further discloses and claims, in part, said resource identifier, in one embodiment, identifies a

request for dynamically generated information wherein said request is mapped by a thread designated to handle said request. *See* '425 Patent at 22:1-56 and claims 2, 4 , 14 and 24.

16. The '425 Patent further discloses and claims, in part, said receiver apparatus comprising means for receiving program-related signals for receiving said hyperlink address string and means for processing for processing said hyperlink address string wherein said means for receiving program-related signals comprises timer means, receives and processes said hyperlink address string structured as a PrintWriter method to process A) said predetermined hyperlink address comprising said resource identifier to hyperlink to said resource in the initial array position of said list in which resource identifiers uniquely identifying resources corresponding to predetermined program material are arrayed and B) said one or more parameters defining predetermined printable output of said resource in the initial array position of a list in which resource identifiers uniquely identifying resources corresponding to predetermined program material are arrayed comprising at least one parameter instructing a PrintWriter to print said predetermined printable output of said resource in the initial array position of a list in which resource identifiers uniquely identifying resources corresponding to predetermined program material are arrayed. *See* '425 Patent at 1:51-65, 3:33-41, 15:27-35 and 21:6-10 and claims 1, 2, 4, 12, 14 and 18.

17. The '425 Patent is valid and enforceable.

DEDENDANT GOOGLE'S INFRINGING BROADCASTRECEIVER

18. Defendant Google operates a mobile communications infrastructure called the Android Platform connecting to applications from applications developers and devices from device manufacturers. The Android Platform includes a software development kit for developing

Android applications and an operating system featuring the "Android Runtime" for applications to run on, *inter alia*, the Android BroadcastReceiver Component system.

19. In Defendant Google's BroadcastReceiver Component system, a virtual machine comprises either a transmit JVM or VM, for example at a server, or a receive JVM or VM, for example at a mobile receiver or set-top box wherein each said transmit and receive JVM or VM work together to communicate and interact with each other using Java code in the Java Programming Language including a list of one or more print() and println() statements of the PrintWriter method wherein said list of one or more print() and println() statements of the PrintWriter method is generated and encoded to be transmitted in conjunction with program signals representative of predetermined program material typically at said transmit JVM or VM and received and processed at a receiver apparatus including said receive JVM or VM.

20. Defendant Google deploys certain source (.java) code containing, *inter alia*, certain PrintWriter methods and respective lists of one or more print() or println() statements of the PrintWriter method having a pw or an out parameter in the Android BroadcastReceiver Component system. *See* generally https://android.googlesource.com/platform/frameworks/base.git/+/jb-mr2-release/core/java/android/content/BroadcastReceiver.java.

21. The Android operating system software deploys a "stack" consisting of Java applications running on a java-based object oriented application framework and core libraries running on the Android Runtime that features ahead-of-time (AOT) compilation.

22. Defendant Google actively distributes said Android BroadcastReceiver Component system and promotes its use in applications by third party developers and manufacturers of receiving devices using the Android Runtime.

23. Through said Android BroadcastReceiver Component, Defendant Google distributes certain program signals representative of predetermined program material comprising one or more print() or println() statements of the PrintWriter method documented below (hereinafter termed "**Defendant Google's Infringing Code**") to be transmitted to and received and processed by an apparatus comprising the Android Runtime to hyperlink to predetermined content from said one or more print() or println() statements of the PrintWriter method.

24. In said Android BroadcastReceiver Component, Defendant Google generates and/or encodes **Defendant Google's Infringing Code** to be transmitted in conjunction with said certain program signals representative of predetermined program material and received and processed by an apparatus comprising the Android Runtime to hyperlink to resources comprising said one or more print() or println() statements of the PrintWriter method corresponding to predetermined program material.

25. Said program signals representative of predetermined program material transmitted via Defendant Google's Android BroadcastReceiver Component system comprise a predetermined announcement (program material) corresponding to a predetermined "intent" programmed into Android BroadcastReceiver Component system receivers comprising the Android Runtime and when a system or application event occurs, registered BroadcastReceiver receivers respond. *See*

https://android.googlesource.com/platform/frameworks/base.git/+/master/core/java/android/content/Intent.java.

26. **Defendant Google's Infringing Code** containing said print() or println() statements of the PrintWriter method to be transmitted in conjunction with program signals representative of predetermined program material via Defendant Google's Android BroadcastReceiver service comprises a first attribute indicating a predetermined hyperlink address comprising a resource identifier identifying a resource in the initial array position of a list in which resource identifiers uniquely identifying resources corresponding to said program signals representative of predetermined program material transmitted via Defendant Google's Android BroadcastReceiver service are arrayed and a second attribute defining predetermined printable output of said resource in the initial array position instructing a PrintWriter to print said predetermined printable output of said resource in the initial array position.

27. Said print() or println() statements of the PrintWriter method to be generated and/or transmitted in conjunction with program signals representative of predetermined program material via Defendant Google's Android BroadcastReceiver service comprise a set of statements to be used directly or indirectly by a PrintWriter method of operation by which underlying program code is operated and controlled and essential to achieving program interoperability in a virtual machine environment, enabling developers to write code based on and to format a predetermined set of arguments in an argumentList defined within a print() or println() statement of the PrintWriter method used to deploy applications for intended statement return.

28. Any one or more print() or println() statements of the PrintWriter method copied in this Complaint, specifically **Defendant Google's Infringing Code** documented below, is strictly for one or more purposes of any legal proceeding involving or otherwise related to this Complaint.

DEFENDANT GOOGLE'S INFRINGING CODE

29. Said print() or println() statements of the PrintWriter method contained in Defendant Google's infringing code specifically comprise the following print() or println() statements of the PrintWriter method having a pw parameter:

```
                pw.println("No process found for: " + args[0]);
                                        …
   pw.print(mActivityManagerService.mProcessStats.printCurrentState());
                                        …
pw.println("Permission Denial: can't dump ActivityManager from from
pid="

            + Binder.getCallingPid()

            + ", uid=" + Binder.getCallingUid()

            + " without permission "

            + android.Manifest.permission.DUMP);
                                        …
        pw.println("Activity manager dump options:");

        pw.println("  [-a] [-h] [cmd] ...");

        pw.println("  cmd may be one of:");

        pw.println("    activities: activity stack state");

        pw.println("    broadcasts: broadcast state");

        pw.println("    intents: pending intent state");

        pw.println("    processes: process state");

        pw.println("    providers: content provider state");
```

```
                pw.println("    services: service state");

                pw.println("    service [name]: service client-side state");
                                        …
                pw.println("Unknown argument: " + opt + "; use -h for help");
                                        …
                pw.println("Providers in Current Activity Manager State:");
                                        …
                pw.println(" ");
                                        …
                pw.println("------------------------------------------------
--------------------------------");

                pw.println("Broadcasts in Current Activity Manager State:");
                                        …
                pw.println(" ");
                                        …
                pw.println("------------------------------------------------
--------------------------------");

                pw.println("Services in Current Activity Manager State:");
                                        …
                pw.println(" ");
                                        …
                pw.println("------------------------------------------------
--------------------------------");

                pw.println("PendingIntents in Current Activity Manager
State:");
                                        …
                pw.println(" ");
                                        …
                pw.println("------------------------------------------------
--------------------------------");

                pw.println("Activities in Current Activity Manager State:");
                                        …
                pw.println(" ");
                                        …

                pw.println("------------------------------------------------
--------------------------------");
```

```
                pw.println("Processes in Current Activity Manager State:");
                                                …
            pw.println("  Activity stack:");
                                                …
        pw.println(" ");

        pw.println("  Running activities (most recent first):");
                                                …
            pw.println(" ");

            pw.println("  Activities waiting for another to become
visible:");
                                                …
            pw.println(" ");

            pw.println("  Activities waiting to stop:");
                                                …
            pw.println(" ");

            pw.println("  Activities waiting to finish:");
                                                …
        pw.println(" ");

        pw.println("  mPausingActivity: " + mPausingActivity);

        pw.println("  mResumedActivity: " + mResumedActivity);

        pw.println("  mFocusedActivity: " + mFocusedActivity);

        pw.println("  mLastPausedActivity: " + mLastPausedActivity);
                                                …
            pw.println(" ");

            pw.println("Recent tasks in Current Activity Manager State:");
                                                …
                pw.print("  * Recent #"); pw.print(i); pw.print(": ");

                        pw.println(tr);
                                                …
        pw.println(" ");

        pw.println("  mCurTask: " + mCurTask);
                                                …
```

```
                    pw.println("  All known processes:");
                                        …
                pw.print(r.persistent ? "  *PERS*" : "  *APP*");
                    pw.print(" UID "); pw.print(procs.keyAt(ia));
                    pw.print(" "); pw.println(r);
                                        …
                     pw.println(" ");
                                        …
            pw.println("  Running processes (most recent first):");
                                        …
                        pw.println(" ");
                                        …
                pw.println("  PID mappings:");
                                        …
                    pw.print("    PID #");
pw.print(mPidsSelfLocked.keyAt(i));
                    pw.print(": ");
pw.println(mPidsSelfLocked.valueAt(i));
                                        …
                     pw.println(" ");
                                        …
            pw.println("  Foreground Processes:");
                                        …
                pw.print("    PID #");
pw.print(mForegroundProcesses.keyAt(i));
                    pw.print(": ");
pw.println(mForegroundProcesses.valueAt(i));
                                        …
                     pw.println(" ");
                                        …
            pw.println("  Persisent processes that are starting:");
                                        …
                     pw.println(" ");
                                        …
            pw.println("  Processes that are starting:");
                                        …
                      pw.println(" ");
```

```
                                    …
            pw.println("  Processes that are being removed:");
                                    …
                         pw.println(" ");
                                    …
            pw.println("  Processes that are on old until the system is
ready:");
                                    …
                        pw.println(" ");
                                    …
            pw.println("  Processes that are waiting to GC:");
                                    …
                pw.print("    Process "); pw.println(proc);
                pw.print("      lowMem="); pw.print(proc.reportLowMemory);
                        pw.print(", last gced=");
                        pw.print(now-proc.lastRequestedGc);
                        pw.print(" ms ago, last lowMem=");
                        pw.print(now-proc.lastLowMemory);
                        pw.println(" ms ago");
                                    …
                         pw.println(" ");
                                    …
            pw.println("  Time since processes crashed:");
                                    …
                    pw.print("    Process "); pw.print(procs.getKey());
                            pw.print(" uid "); pw.print(uids.keyAt(i));
                            pw.print(": last crashed ");
                            pw.print((now-uids.valueAt(i)));
                            pw.println(" ms ago");
                                    …
                        pw.println(" ");
                                    …
            pw.println("  Bad processes:");
                                    …
```

```
                    pw.print("    Bad process "); pw.print(procs.getKey());
                            pw.print(" uid "); pw.print(uids.keyAt(i));
                            pw.print(": crashed at time ");
                            pw.println(uids.valueAt(i));
                                    …
        pw.println(" ");
        pw.println("  mHomeProcess: " + mHomeProcess);
        pw.println("  mConfiguration: " + mConfiguration);
        pw.println("  mConfigWillChange: " + mConfigWillChange);
        pw.println("  mSleeping=" + mSleeping + " mShuttingDown=" +
mShuttingDown);
                                    …
            pw.println("  mDebugApp=" + mDebugApp + "/orig=" + mOrigDebugApp
                    + " mDebugTransient=" + mDebugTransient
                    + " mOrigWaitForDebugger=" + mOrigWaitForDebugger);
                                    …
            pw.println("  mAlwaysFinishActivities=" + mAlwaysFinishActivities
                    + " mController=" + mController);
                                    …
            pw.println("  Total persistent processes: " + numPers);
            pw.println("  mStartRunning=" + mStartRunning
                    + " mSystemReady=" + mSystemReady
                    + " mBooting=" + mBooting
                    + " mBooted=" + mBooted
                    + " mFactoryTest=" + mFactoryTest);
            pw.println("  mGoingToSleep=" + mGoingToSleep);
            pw.println("  mLaunchingActivity=" + mLaunchingActivity);
            pw.println("  mAdjSeq=" + mAdjSeq + " mLruSeq=" + mLruSeq);
```

```
                                    …
        pw.println("  Service " + r.name.flattenToString());
                                    …
                pw.print("\n");
                                    …
                pw.println("got a RemoteException while dumping the
service");
                                    …
                pw.println(" ");
                pw.println("  Registered Receivers:");
                                    …
                    pw.print("  * "); pw.println(r);
                                    …
            pw.println(" ");
            pw.println("Receiver Resolver Table:");
                                    …
                pw.println(" ");
                pw.println("  Active broadcasts:");
                                    …
                pw.println("  Broadcast #" + i + ":");
                                    …
                pw.println(" ");
                pw.println("  Active ordered broadcasts:");
                                    …
                pw.println("  Serialized Broadcast #" + i + ":");
                                    …
            pw.println(" ");
            pw.println("  Pending broadcast:");
                                    …
                pw.println("    (null)");
                                    …
            pw.println(" ");
            pw.println("  Historical broadcasts:");
                                    …
                pw.println("  Historical Broadcast #" + i + ":");
```

```
                                …
pw.println(" ");

pw.println("  Sticky broadcasts:");
                                …
    pw.print("  * Sticky action "); pw.print(ent.getKey());

            pw.println(":");
                                …
        pw.println(sb.toString());
                                …

            pw.print("      ");

            pw.println(bundle.toString());
                                …
pw.println(" ");

pw.println("  mBroadcastsScheduled=" + mBroadcastsScheduled);

pw.println("  mHandler:");
                                …

    pw.println("  Active services:");
                                …
        pw.print("  * "); pw.println(r);
                                …
pw.println("  Pending services:");
                                …
    pw.print("  * Pending "); pw.println(r);
                                …
             pw.println(" ");

pw.println("  Restarting services:");
                                …
    pw.print("  * Restarting "); pw.println(r);
                                …
             pw.println(" ");

pw.println("  Stopping services:");
                                …
    pw.print("  * Stopping "); pw.println(r);
                                …
                pw.println(" ");
```

```
        pw.println("  Connection bindings to services:");
                                …
            pw.print("  * "); pw.println(r);
                                …
                    pw.println(" ");
        pw.println("  Published content providers (by class):");
                                …
            pw.print("  * "); pw.println(r);
                                …
        pw.println(" ");
        pw.println("  Authority to provider mappings:");
                                …
            pw.print("  "); pw.print(e.getKey()); pw.print(": ");
                    pw.println(r);
                                …
                pw.println(" ");
    pw.println("  Launching content providers:");
                                …
        pw.print("  Launching #"); pw.print(i); pw.print(": ");
                pw.println(mLaunchingProviders.get(i));
                                …
    pw.println();
    pw.println("Granted Uri Permissions:");
                                …
        pw.print("  * UID "); pw.print(uid);
                pw.println(" holds:");
                                …
            pw.print("    "); pw.println(perm);
                                …
                pw.print("  * "); pw.println(rec);
                                …
                pw.print("  * "); pw.print(ref);
```

```
                                    …
                pw.print(prefix);

                pw.print(full ? "* " : "  ");

                pw.println(lastTask);
                                    …
            pw.print(prefix); pw.print(full ? "  * " : "    ");
pw.print(label);

            pw.print(" #"); pw.print(i); pw.print(": ");

            pw.println(r);
                                    …
                pw.println(prefix + (r.persistent ? persistentLabel :
normalLabel)

                      + " #" + i + ":");
                                    …
                pw.println(String.format("%s%s #%2d: adj=%s/%s %s (%s)",

                        prefix, (r.persistent ? persistentLabel :
normalLabel),

                        i, oomAdj, schedGroup, r.toShortString(),
r.adjType));
                                    …
                    pw.println(prefix + "          " + r.adjTarget

                            + "<=" + r.adjSource);
                                    …
                pw.println(String.format("%s%s #%2d: %s",

                        prefix, (r.persistent ? persistentLabel :
normalLabel),

                        i, r.toString()));
                                    …
            pw.println(uptime + "," + realtime);
                                    …
            pw.println("Applications Memory Usage (kB):");
```

```
            pw.println("Uptime: " + uptime + " Realtime: " + realtime);
                                        …
                    pw.println("\n** MEMINFO in pid " + r.pid + " [" +
r.processName + "] **");
                                        …
                        pw.println("Got RemoteException!");
                                        … .
```

See https://android.googlesource.com/platform/frameworks/base/+/refs/heads/froyo-release/services/java/com/android/server/am/ActivityManagerService.java.

30. Said print() or println() statements of the PrintWriter method contained in Defendant Google's infringing code specifically comprise the following print() or println() statements of the PrintWriter method having a pw parameter:

```
pw.println("Permission Denial: can't dump AlarmManager from from pid="
                + Binder.getCallingPid()
                + ", uid=" + Binder.getCallingUid());
                                        …
            pw.println("Current Alarm Manager state:");
                                        …
                pw.println(" ");
                pw.print("  Realtime wakeup (now=");
                        pw.print(sdf.format(new Date(now)));
pw.println("):");
                                        …
();
                pw.println(" ");
                pw.print("  Elapsed realtime wakeup (now=");
                                        …
                                                        pw.println("):");
                                        …
```

```
            pw.println();

            pw.print("  Broadcast ref count: ");
pw.println(mBroadcastRefCount);

            pw.println();
                                    …

                pw.println();

                                    …

                pw.println("  Top Alarms:");

                                   ...

                    pw.print("    ");
                                    …

                                      pw.print("*ACTIVE* ");

                                    …

                    pw.print(" running, "); pw.print(fs.numWakeup);

                    pw.print(" wakeups, "); pw.print(fs.count);

                    pw.print(" alarms: ");
pw.print(fs.mBroadcastStats.mPackageName);

                    pw.println();

                    pw.print("      ");

                                    …

                        pw.print(" act="); pw.print(fs.mTarget.first);

                                    …

                        pw.print(" cmp=");
pw.print(fs.mTarget.second.toShortString());

                                    …
                    pw.println();
                                    …
```

```
            pw.println(" ");
            pw.println("  Alarm Stats:");
                                    …
                pw.print("  ");
                                    …
               pw.print("*ACTIVE* ");
                pw.print(be.getKey());
                pw.print(" "); TimeUtils.formatDuration(bs.aggregateTime,
pw);
                        pw.print(" running, "); pw.print(bs.numWakeup);
                        pw.println(" wakeups:");
                                    …
                    pw.print("    ");
                            …
                            pw.print(" "); pw.print(fs.numWakeup);
                            pw.print(" wakes " ); pw.print(fs.count);
                            pw.print(" alarms:");
                                    …
                                pw.print(" act=");
pw.print(fs.mTarget.first);
                                    …
                                pw.print(" cmp=");
pw.print(fs.mTarget.second.toShortString());
                                    …
                            pw.println();
                                    …
            pw.print(prefix); pw.print(label); pw.print(" #"); pw.print(i);
                    pw.print(": "); pw.println(a);
                                    …
```

```
            pw.print(prefix); pw.print("type="); pw.print(type);
                    pw.print(" when="); TimeUtils.formatDuration(when, now,
pw);
                    pw.print(" repeatInterval="); pw.print(repeatInterval);
                    pw.print(" count="); pw.println(count);
            pw.print(prefix); pw.print("operation="); pw.println(operation);
                                        … .
```

See https://android.googlesource.com/platform/frameworks/base.git/+/android-4.2.2_r1/services/java/com/android/server/AlarmManagerService.java.

31. Said print() or println() statements of the PrintWriter method contained in Defendant Google's infringing code specifically comprise the following print() or println() statements of the PrintWriter method having a pw parameter:

```
            pw.print("   Muted: ");
            pw.println(mIsMuted);
            pw.print("   Min: ");
            pw.println((mIndexMin + 5) / 10);
            pw.print("   Max: ");
            pw.println((mIndexMax + 5) / 10);
            pw.print("   Current: ");
                                    …
                    pw.print(", ");
                                    …
                pw.print(Integer.toHexString(device));
                                    …
                    pw.print(" (");
                    pw.print(deviceName);
                    pw.print(")");
                                    …
```

```
                pw.print(": ");
                                        …
                pw.print(index);
                                        …
            pw.println();
            pw.print("   Devices: ");
                                        …
                        pw.print(", ");
                                        …
                    pw.print(AudioSystem.getOutputDeviceName(device));
                                          ….
```

See https://android.googlesource.com/platform/frameworks/base/+/master/services/core/java/com/android/server/audio/AudioService.java.

32. Said print() or println() statements of the PrintWriter method contained in Defendant Google's infringing code specifically comprise the following print() or println() statements of the PrintWriter method having a pw parameter:

```
pw.print("Battery: stable power = ");
        pw.print(mChargeTracker.isOnStablePower());
        pw.print(", not low = ");
        pw.println(mChargeTracker.isBatteryNotLow());
                                        …
        if (mChargeTracker.isMonitoring()) {
            pw.print("MONITORING: seq=");
            pw.println(mChargeTracker.getSeq());
                                        …
        pw.print("Tracking ");
        pw.print(mTrackedTasks.size());
        pw.println(":");
        for (int i = 0; i < mTrackedTasks.size(); i++) {
```

```
            final JobStatus js = mTrackedTasks.valueAt(i);
            if (!js.shouldDump(filterUid)) {
                continue;
            }
            pw.print("  #");
            js.printUniqueId(pw);
            pw.print(" from ");
            UserHandle.formatUid(pw, js.getSourceUid());
            pw.println();
                                        … .
```

See https://android.googlesource.com/platform/frameworks/base/+/master/services/core/java/com/android/server/job/controllers/BatteryController.java?autodive=0%2F%2F%2F%2F%2F%2F%2F.

33. Said print() or println() statements of the PrintWriter method contained in Defendant Google's infringing code specifically comprise the following print() or println() statements of the PrintWriter method having a pw parameter:

```
pw.println("Battery service (battery) commands:");
        pw.println("  help");
        pw.println("    Print this help text.");
        pw.println("  set [-f]
[ac|usb|wireless|status|level|temp|present|invalid] <value>");
        pw.println("    Force a battery property value, freezing battery
state.");
```

```
        pw.println("    -f: force a battery change broadcast be sent, prints
new sequence.");

        pw.println("  unplug [-f]");

        pw.println("    Force battery unplugged, freezing battery state.");

        pw.println("    -f: force a battery change broadcast be sent, prints
new sequence.");

        pw.println("  reset [-f]");

        pw.println("    Unfreeze battery state, returning to current hardware
values.");

        pw.println("    -f: force a battery change broadcast be sent, prints
new sequence.");
                                        …
pw.println("No property specified");
                                        …
                    pw.println("No value specified");
                                        …

     pw.println("Bad value: " + value);

                                         …

pw.println(mSequence);
                                        …
                pw.println("Current Battery Service state:");
                                        …
                    pw.println("  (UPDATES STOPPED -- use 'reset' to
restart)");
                                        …
                pw.println("  AC powered: " + mBatteryProps.chargerAcOnline);

                pw.println("  USB powered: " +
mBatteryProps.chargerUsbOnline);

                pw.println("  Wireless powered: " +
mBatteryProps.chargerWirelessOnline);

                pw.println("  Max charging current: " +
mBatteryProps.maxChargingCurrent);
```

```
                pw.println("  Max charging voltage: " +
mBatteryProps.maxChargingVoltage);

                pw.println("  Charge counter: " +
mBatteryProps.batteryChargeCounter);

                pw.println("  status: " + mBatteryProps.batteryStatus);

                pw.println("  health: " + mBatteryProps.batteryHealth);

                pw.println("  present: " + mBatteryProps.batteryPresent);

                pw.println("  level: " + mBatteryProps.batteryLevel);

                pw.println("  scale: " + BATTERY_SCALE);

                pw.println("  voltage: " + mBatteryProps.batteryVoltage);

                pw.println("  temperature: " +
mBatteryProps.batteryTemperature);

                pw.println("  technology: " +
mBatteryProps.batteryTechnology);
                                        … .
```

See https://android.googlesource.com/platform/frameworks/base/+/master/services/core/java/com/android/server/BatteryService.java.

34. Said print() or println() statements of the PrintWriter method contained in Defendant Google's infringing code specifically comprise the following print() or println() statements of the PrintWriter method having a pw parameter:

```
public void dump(FileDescriptor fd, PrintWriter pw, String[] args) {

            pw.println("Carrier app binding for phone " + phoneId);

            pw.println("  connection: " + connection);

            pw.println("  bindCount: " + bindCount);

            pw.println("  lastBindStartMillis: " + lastBindStartMillis);
```

```
            pw.println("  unbindCount: " + unbindCount);

            pw.println("  lastUnbindMillis: " + lastUnbindMillis);

            pw.println("  mUnbindScheduledUptimeMillis: " +
mUnbindScheduledUptimeMillis);

            pw.println();
                                    …
        pw.println("CarrierServiceBindHelper:");
                                    … .
```

See https://android.googlesource.com/platform/frameworks/opt/telephony/+/master/src/java/com/android/internal/telephony/CarrierServiceBindHelper.java.

35. Said print() or println() statements of the PrintWriter method contained in Defendant Google's infringing code specifically comprise the following print() or println() statements of the PrintWriter method having a pw parameter:

```
pw.println(String.format(

                        "Permission Denial: can't dump CommonTimeManagement
service from from " +

                        "pid=%d, uid=%d", Binder.getCallingPid(),
Binder.getCallingUid()));
                                    …
            pw.println("Native Common Time service was not detected at
startup.  " +

                       "Service is unavailable");
                                    …
            pw.println("Current Common Time Management Service Config:");

            pw.println(String.format("  Native service     : %s",

                                    (null == mCTConfig) ? "reconnecting"

                                                        : "alive"));

            pw.println(String.format("  Bound interface    : %s",
```

```
                                    (null == mCurIface ? "unbound" :
mCurIface)));

            pw.println(String.format("  Allow WiFi         : %s", ALLOW_WIFI
? "yes" : "no"));

            pw.println(String.format("  Allow Auto Disable : %s",
AUTO_DISABLE ? "yes" : "no"));

            pw.println(String.format("  Server Priority    : %d",
mEffectivePrio));

            pw.println(String.format("  No iface timeout   : %d",
NO_INTERFACE_TIMEOUT));
                                    … .
```

See https://chromium.googlesource.com/android_tools/+/refs/heads/master/sdk/sources/android-25/com/android/server/CommonTimeManagementService.java.

36. Said print() or println() statements of the PrintWriter method contained in Defendant Google's infringing code specifically comprise the following print() or println() statements of the PrintWriter method having a pw parameter:

```
pw.println("  Settings:");


            pw.print("    ");
pw.print(KEY_LIGHT_IDLE_AFTER_INACTIVE_TIMEOUT); pw.print("=");
                                    …
            pw.println();


            pw.print("    "); pw.print(KEY_LIGHT_PRE_IDLE_TIMEOUT);
pw.print("=");
                                    …
            pw.println();


            pw.print("    "); pw.print(KEY_LIGHT_IDLE_TIMEOUT);
pw.print("=");
```

```
                                        …
            pw.println();


            pw.print("    "); pw.print(KEY_LIGHT_IDLE_FACTOR); pw.print("=");
            pw.print(LIGHT_IDLE_FACTOR);
            pw.println();


            pw.print("    "); pw.print(KEY_LIGHT_MAX_IDLE_TIMEOUT);
pw.print("=");
                                        …
            pw.println();


            pw.print("    ");
pw.print(KEY_LIGHT_IDLE_MAINTENANCE_MIN_BUDGET); pw.print("=");
                                        …
            pw.println();


            pw.print("    ");
pw.print(KEY_LIGHT_IDLE_MAINTENANCE_MAX_BUDGET); pw.print("=");
                                        …
            pw.println();


            pw.print("    "); pw.print(KEY_MIN_LIGHT_MAINTENANCE_TIME);
pw.print("=");
                                        …
            pw.println();


            pw.print("    "); pw.print(KEY_MIN_DEEP_MAINTENANCE_TIME);
pw.print("=");
                                        …
            pw.println();
```

```
            pw.print("    "); pw.print(KEY_INACTIVE_TIMEOUT); pw.print("=");
                                        …
            pw.println();


            pw.print("    "); pw.print(KEY_SENSING_TIMEOUT); pw.print("=");
                                        …
            pw.println();


            pw.print("    "); pw.print(KEY_LOCATING_TIMEOUT); pw.print("=");
                                        …
            pw.println();


            pw.print("    "); pw.print(KEY_LOCATION_ACCURACY); pw.print("=");
            pw.print(LOCATION_ACCURACY); pw.print("m");
            pw.println();


            pw.print("    "); pw.print(KEY_MOTION_INACTIVE_TIMEOUT);
pw.print("=");
                                        …
            pw.println();


            pw.print("    "); pw.print(KEY_IDLE_AFTER_INACTIVE_TIMEOUT);
pw.print("=");
                                        …
            pw.println();


            pw.print("    "); pw.print(KEY_IDLE_PENDING_TIMEOUT);
pw.print("=");
```

```
                                        …
            pw.println();


            pw.print("    "); pw.print(KEY_MAX_IDLE_PENDING_TIMEOUT);
pw.print("=");
                                        …
            pw.println();


            pw.print("    "); pw.print(KEY_IDLE_PENDING_FACTOR);
pw.print("=");
            pw.println(IDLE_PENDING_FACTOR);


            pw.print("    "); pw.print(KEY_IDLE_TIMEOUT); pw.print("=");
                                        …
            pw.println();


            pw.print("    "); pw.print(KEY_MAX_IDLE_TIMEOUT); pw.print("=");
                                        …
            pw.println();


            pw.print("    "); pw.print(KEY_IDLE_FACTOR); pw.print("=");
            pw.println(IDLE_FACTOR);


            pw.print("    "); pw.print(KEY_MIN_TIME_TO_ALARM); pw.print("=");
                                        …
            pw.println();


            pw.print("    "); pw.print(KEY_MAX_TEMP_APP_WHITELIST_DURATION);
pw.print("=");
```

```
                                        …
            pw.println();


            pw.print("    "); pw.print(KEY_MMS_TEMP_APP_WHITELIST_DURATION);
pw.print("=");
                                        …
            pw.println();


            pw.print("    "); pw.print(KEY_SMS_TEMP_APP_WHITELIST_DURATION);
pw.print("=");
                                        …

            pw.println();


            pw.print("    "); pw.print(KEY_NOTIFICATION_WHITELIST_DURATION);
pw.print("=");
                                        …
            pw.println();
                                          …

pw.println("Device idle controller (deviceidle) commands:");

        pw.println("  help");

        pw.println("    Print this help text.");

        pw.println("  step [light|deep]");

        pw.println("    Immediately step to next state, without waiting for
alarm.");

        pw.println("  force-idle [light|deep]");

        pw.println("    Force directly into idle mode, regardless of other
device state.");

        pw.println("  force-inactive");
```

```
        pw.println("    Force to be inactive, ready to freely step idle
states.");

        pw.println("  unforce");

        pw.println("    Resume normal functioning after force-idle or force-
inactive.");

        pw.println("  get [light|deep|force|screen|charging|network]");

        pw.println("    Retrieve the current given state.");

        pw.println("  disable [light|deep|all]");

        pw.println("    Completely disable device idle mode.");

        pw.println("  enable [light|deep|all]");

        pw.println("    Re-enable device idle mode after it had previously
been disabled.");

        pw.println("  enabled [light|deep|all]");

        pw.println("    Print 1 if device idle mode is currently enabled,
else 0.");

        pw.println("  whitelist");

        pw.println("    Print currently whitelisted apps.");

        pw.println("  whitelist [package ...]");

        pw.println("    Add (prefix with +) or remove (prefix with -)
packages.");

        pw.println("  except-idle-whitelist [package ...|reset]");

        pw.println("    Prefix the package with '+' to add it to whitelist or
"

                + "'=' to check if it is already whitelisted");

        pw.println("    [reset] will reset the whitelist to it's original
state");

        pw.println("    Note that unlike <whitelist> cmd, "
```

```
                + "changes made using this won't be persisted across boots");

        pw.println("  tempwhitelist");

        pw.println("    Print packages that are temporarily whitelisted.");

        pw.println("  tempwhitelist [-u USER] [-d DURATION] [package ..]");

        pw.println("    Temporarily place packages in whitelist for DURATION
milliseconds.");

        pw.println("    If no DURATION is specified, 10 seconds is used");
                                        ...

pw.println("Device idle controller (deviceidle) commands:");

        pw.println("  help");

        pw.println("    Print this help text.");

        pw.println("  step [light|deep]");

        pw.println("    Immediately step to next state, without waiting for
alarm.");

        pw.println("  force-idle [light|deep]");

        pw.println("    Force directly into idle mode, regardless of other
device state.");

        pw.println("  force-inactive");

        pw.println("    Force to be inactive, ready to freely step idle
states.");

        pw.println("  unforce");

        pw.println("    Resume normal functioning after force-idle or force-
inactive.");

        pw.println("  get [light|deep|force|screen|charging|network]");

        pw.println("    Retrieve the current given state.");

        pw.println("  disable [light|deep|all]");
```

```
        pw.println("    Completely disable device idle mode.");

        pw.println("  enable [light|deep|all]");

        pw.println("    Re-enable device idle mode after it had previously
been disabled.");

        pw.println("  enabled [light|deep|all]");

        pw.println("    Print 1 if device idle mode is currently enabled,
else 0.");

        pw.println("  whitelist");

        pw.println("    Print currently whitelisted apps.");

        pw.println("  whitelist [package ...]");

        pw.println("    Add (prefix with +) or remove (prefix with -)
packages.");

        pw.println("  except-idle-whitelist [package ...|reset]");

        pw.println("    Prefix the package with '+' to add it to whitelist or
"

                + "'=' to check if it is already whitelisted");

        pw.println("    [reset] will reset the whitelist to it's original
state");

        pw.println("    Note that unlike <whitelist> cmd, "

                + "changes made using this won't be persisted across boots");

        pw.println("  tempwhitelist");

        pw.println("    Print packages that are temporarily whitelisted.");

        pw.println("  tempwhitelist [-u USER] [-d DURATION] [package ..]");

        pw.println("    Temporarily place packages in whitelist for DURATION
milliseconds.");

        pw.println("If no DURATION is specified, 10 seconds is used");
                                    ...
```

```
pw.print("Stepped to deep: ");

                        pw.println(stateToString(mState));
                                    …
                        pw.print("Stepped to light: ");
pw.println(lightStateToString(mLightState));
                                    …
                        pw.println("Unknown idle mode: " + arg);
pw.print("Unable to go deep idle; stopped at ");

                                pw.println(stateToString(mState));

                                exitForceIdleLocked();

                                return -1;
                                    …
                        pw.println("Now forced in to deep idle mode");
                                    …
                                pw.print("Unable to go light idle; stopped at
");

                                pw.println(lightStateToString(mLightState));
                                    …
                        pw.println("Now forced in to light idle mode");
                                    …
                        pw.println("Unknown idle mode: " + arg);
                                    …
                    pw.print("Light state: ");

                    pw.print(lightStateToString(mLightState));

                    pw.print(", deep state: ");

                    pw.println(stateToString(mState));
                                    …
                    pw.print("Light state: ");

                    pw.print(lightStateToString(mLightState));

                    pw.print(", deep state: ");

                    pw.println(stateToString(mState));
                                    …
                                        pw.println(stateToString(mState));
```

```
                                    …
                                        pw.println(mForceIdle);
                                    …
                                         pw.println(mScreenOn);
                                    …
                                           pw.println(mCharging);
                                    …
                                          pw.println(mNetworkConnected);
                                    …
                                   pw.println("Unknown get option: " +
arg);
                                    …
                        pw.println("Argument required");
                                    …
                                pw.println("Deep idle mode disabled");
                                    …
                                pw.println("Light idle mode disabled");
                                    …
                            pw.println("Unknown idle mode: " + arg);
                                    …
                                pw.println("Deep idle mode enabled");
                                    …
                                pw.println("Light idle mode enable");
                                    …
                            pw.println("Unknown idle mode: " + arg);
                                    …
                        pw.println(mDeepEnabled && mLightEnabled ? "1" : 0);
                                    …
                        pw.println(mDeepEnabled ? "1" : 0);
                                    …
                        pw.println(mLightEnabled ? "1" : 0);
                                    …
                        pw.println("Unknown idle mode: " + arg);
                                    …
                                pw.println("Package must be prefixed with +, -,
or =: " + arg);
                                    …
                                pw.println("Added: " + pkg);
                                    …
                                pw.println("Unknown package: " + pkg);
```

```
                                        …
                                    pw.println("Removed: " + pkg);
                                        …
                                pw.print("system-excidle,");
                                pw.print(mPowerSaveWhitelistAppsExceptIdle.keyAt(j));
                                pw.print(",");
pw.println(mPowerSaveWhitelistAppsExceptIdle.valueAt(j));
                                        …
                                pw.print("system,");
                                pw.print(mPowerSaveWhitelistApps.keyAt(j));
                                pw.print(",");
                                pw.println(mPowerSaveWhitelistApps.valueAt(j));
                                        …
                                pw.print("user,");
                                pw.print(mPowerSaveWhitelistUserApps.keyAt(j));
                                pw.print(",");
                                pw.println(mPowerSaveWhitelistUserApps.valueAt(j));
                                        …
                                pw.println("-u requires a user number");
                                        …
                                pw.println("-d requires a duration");
                                        …
                            pw.println("Failed: " + e);
                                        …
                            pw.println("No arguments given");
                                        …
                                    pw.println("Package must be prefixed with +, -,
or =: " + arg);
                                        …
                                    pw.println("Added: " + pkg);
                                        …
                                    pw.println("Unknown package: " + pkg);
                                        …
```

```
pw.println(g etPowerSaveWhitelistExceptIdleInternal(pkg));
                                    …
                            pw.println("Unknown argument: " + arg);
                                    …
                        pw.println("Unknown option: " + arg);
                                    …
                    pw.println("  Idling history:");
                                    …
                        pw.print("    ");
                        pw.print(label);
                        pw.print(": ");
                                    …
                        pw.println();
                                    …
                    pw.println("  Whitelist (except idle) system apps:");
                                    …
                        pw.print("    ");
                        pw.println(mPowerSaveWhitelistAppsExceptIdle.keyAt(i));
                                    …
                    pw.println("  Whitelist system apps:");
                                    …
                        pw.print("    ");
                        pw.println(mPowerSaveWhitelistApps.keyAt(i));
                                    …
                    pw.println("  Whitelist user apps:");
                                    …
                        pw.print("    ");
                        pw.println(mPowerSaveWhitelistUserApps.keyAt(i));
                                    …
                    pw.println("  Whitelist (except idle) all app ids:");
                                    …
                        pw.print("    ");
                        pw.print(mPowerSaveWhitelistExceptIdleAppIds.keyAt(i));
                        pw.println();
                                    …
                    pw.println("  Whitelist user app ids:");
```

```
                                …
                    pw.print("    ");
                    pw.print(mPowerSaveWhitelistUserAppIds.keyAt(i));
                    pw.println();
                                …
                pw.println("  Whitelist all app ids:");
                                …
                    pw.print("    ");
                    pw.print(mPowerSaveWhitelistAllAppIds.keyAt(i));
                    pw.println();
                                …
                pw.println("  Temp whitelist app ids:");
                                …
                    pw.print("    ");
                    pw.print(mTempWhitelistAppIdArray[i]);
                    pw.println();
                                …
            pw.print("  mLightEnabled="); pw.print(mLightEnabled);
            pw.print("  mDeepEnabled="); pw.println(mDeepEnabled);
            pw.print("  mForceIdle="); pw.println(mForceIdle);
            pw.print("  mMotionSensor="); pw.println(mMotionSensor);
            pw.print("  mScreenOn="); pw.println(mScreenOn);
            pw.print("  mNetworkConnected="); pw.println(mNetworkConnected);
            pw.print("  mCharging="); pw.println(mCharging);
            pw.print("  mMotionActive="); pw.println(mMotionListener.active);
            pw.print("  mNotMoving="); pw.println(mNotMoving);
            pw.print("  mLocating="); pw.print(mLocating); pw.print("
mHasGps=");
                    pw.print(mHasGps); pw.print(" mHasNetwork=");
```

```
                    pw.print(mHasNetworkLocation); pw.print(" mLocated=");
pw.println(mLocated);
                                        …
                pw.print("  mLastGenericLocation=");
pw.println(mLastGenericLocation);
                                        …
                pw.print("  mLastGpsLocation=");
pw.println(mLastGpsLocation);
                                        …
            pw.print("  mState="); pw.print(stateToString(mState));

            pw.print(" mLightState=");

            pw.println(lightStateToString(mLightState));

            pw.print("  mInactiveTimeout=");
                                        …
            pw.println();
                                        …
                pw.print("  mActiveIdleOpCount=");
pw.println(mActiveIdleOpCount);
                                        …
                pw.print("  mNextAlarmTime=");
                                        …
                pw.println();
                                        …
                pw.print("  mNextIdlePendingDelay=");
                                        …
                pw.println();
                                        …
                pw.print("  mNextIdleDelay=");
                                        …
                pw.println();
                                        …
                pw.print("  mNextIdleDelay=");
                                        …
                pw.println();
                                        …
                pw.print("  mNextLightAlarmTime=");
                                        …
                pw.println();
```

```
                                …
            pw.print("  mCurIdleBudget=");
                                …
            pw.println();
                                …
            pw.print("  mMaintenanceStartTime=");
                                …
            pw.println();
                                …
            pw.print("  mJobsActive="); pw.println(mJobsActive);
                                …
            pw.print("  mAlarmsActive="); pw.println(mAlarmsActive);
                                …
            pw.println("  Temp whitelist schedule:");
                                …
            pw.print(prefix);
            pw.print("UID=");
            pw.print(mTempWhitelistAppIdEndTimes.keyAt(i));
            pw.print(": ");
                                …
            pw.print(" - ");
            pw.println(entry.second);
                                … .
```

See https://android.googlesource.com/platform/frameworks/base/+/master/services/core/java/com/android/server/DeviceIdleController.java.

37. Said print() or println() statements of the PrintWriter method contained in Defendant Google's infringing code specifically comprise the following print() or println() statements of the PrintWriter method having a pw (or p) parameter:

```
pw.print(prefix); pw.print("uid="); pw.println(getUid());
            pw.print(prefix); pw.println("policies:");
                                …
```

```
                    pw.print(prefix); pw.print("   ");
pw.println(pols.get(i).tag);
                                        …
            pw.print(prefix); pw.print("passwordQuality=0x");

                    pw.println(Integer.toHexString(passwordQuality));

            pw.print(prefix); pw.print("minimumPasswordLength=");

                    pw.println(minimumPasswordLength);

            pw.print(prefix); pw.print("passwordHistoryLength=");

                    pw.println(passwordHistoryLength);

            pw.print(prefix); pw.print("minimumPasswordUpperCase=");

                    pw.println(minimumPasswordUpperCase);

            pw.print(prefix); pw.print("minimumPasswordLowerCase=");

                    pw.println(minimumPasswordLowerCase);

            pw.print(prefix); pw.print("minimumPasswordLetters=");

                    pw.println(minimumPasswordLetters);

            pw.print(prefix); pw.print("minimumPasswordNumeric=");

                    pw.println(minimumPasswordNumeric);

            pw.print(prefix); pw.print("minimumPasswordSymbols=");

                    pw.println(minimumPasswordSymbols);

            pw.print(prefix); pw.print("minimumPasswordNonLetter=");

                    pw.println(minimumPasswordNonLetter);

            pw.print(prefix); pw.print("maximumTimeToUnlock=");

                    pw.println(maximumTimeToUnlock);

            pw.print(prefix); pw.print("maximumFailedPasswordsForWipe=");

                    pw.println(maximumFailedPasswordsForWipe);
```

```
            pw.print(prefix); pw.print("specifiesGlobalProxy=");
                    pw.println(specifiesGlobalProxy);
            pw.print(prefix); pw.print("passwordExpirationTimeout=");
                    pw.println(passwordExpirationTimeout);
            pw.print(prefix); pw.print("passwordExpirationDate=");
                    pw.println(passwordExpirationDate);
                                    …
                pw.print(prefix); pw.print("globalProxySpec=");
                        pw.println(globalProxySpec);
                                    …
                pw.print(prefix); pw.print("globalProxyEclusionList=");
                        pw.println(globalProxyExclusionList);
                                    …
            pw.print(prefix); pw.print("encryptionRequested=");
                    pw.println(encryptionRequested);
            pw.print(prefix); pw.print("disableCamera=");
                    pw.println(disableCamera);
            pw.print(prefix); pw.print("disabledKeyguardFeatures=");
                    pw.println(disabledKeyguardFeatures);
                                    …
pw.println("Permission Denial: can't dump DevicePolicyManagerService from
from pid="
                    + Binder.getCallingPid()
                    + ", uid=" + Binder.getCallingUid());
                                    …
            p.println("Current Device Policy Manager state:");
                                    …
                p.println("  Enabled Device Admins (User " +
policy.mUserHandle + "):");
                                    …
```

```
                        pw.print("  ");
pw.print(ap.info.getComponent().flattenToShortString());

                                pw.println(":");
                                        …
                pw.println(" ");

                pw.print("  mPasswordOwner=");
pw.println(policy.mPasswordOwner);
                                        ….
```

See

https://android.googlesource.com/platform/frameworks/base/+/b267554/services/java/com/android/server/DevicePolicyManagerService.java.

38. Said print() or println() statements of the PrintWriter method contained in Defendant Google's infringing code specifically comprise the following print() or println() statements of the PrintWriter method having a pw parameter:

```
pw.println("Device storage monitor service (devicestoragemonitor)
commands:");

        pw.println("  help");

        pw.println("    Print this help text.");

        pw.println("  force-low [-f]");

        pw.println("    Force storage to be low, freezing storage state.");

        pw.println("    -f: force a storage change broadcast be sent, prints
new sequence.");

        pw.println("  force-not-low [-f]");

        pw.println("    Force storage to not be low, freezing storage
state.");

        pw.println("    -f: force a storage change broadcast be sent, prints
new sequence.");

        pw.println("  reset [-f]");
```

```
        pw.println("    Unfreeze storage state, returning to current real
values.");

        pw.println("    -f: force a storage change broadcast be sent, prints
new sequence.");
                                        …
pw.println("Known volumes:");

            pw.increaseIndent();
                                        …
                pw.println("Default:");
                                        …
                pw.println(uuid + ":");

            }

            pw.increaseIndent();

            pw.printPair("level", State.levelToString(state.level));

            pw.printPair("lastUsableBytes", state.lastUsableBytes);

            pw.println();

            pw.decreaseIndent();

        }

        pw.decreaseIndent();

        pw.println();



        pw.printPair("mSeq", mSeq.get());

        pw.printPair("mForceState", State.levelToString(mForceLevel));

        pw.println();

        pw.println();
                                    … .
```

See

https://android.googlesource.com/platform/frameworks/base/+/master/services/core/java/com/android/server/storage/DeviceStorageMonitorService.java.

39. Said print() or println() statements of the PrintWriter method contained in Defendant Google's infringing code specifically comprise the following print() or println() statements of the PrintWriter method having an pw parameter:

```
pw.print("  mDreaming: "); pw.println(mDreaming);

        pw.print("  mBroadcastReceiverRegistered: ");
pw.println(mBroadcastReceiverRegistered);

        pw.print("  mSigMotionSensor: "); pw.println(mSigMotionSensor);

        pw.print("  mPickupSensor:"); pw.println(mPickupSensor);

        pw.print("  mMaxBrightness: "); pw.println(mMaxBrightness);

        pw.print("  mDisplayStateSupported: ");
pw.println(mDisplayStateSupported);

        pw.print("  mNotificationLightOn: ");
pw.println(mNotificationLightOn);

        pw.print("  mMultipulseCount: "); pw.println(mMultipulseCount);

        pw.print("  mNotificationPulseInterval: ");
pw.println(mNotificationPulseInterval);

        pw.print("  mPowerSaveActive: "); pw.println(mPowerSaveActive);
                                    … .
```

See

https://android.googlesource.com/platform/frameworks/base/+/777f5b2/packages/SystemUI/src/com/android/systemui/doze/DozeService.java.

40. Said print() or println() statements of the PrintWriter method contained in Defendant Google's infringing code specifically comprise the following print() or println() statements of the PrintWriter method having an pw parameter:

```
pw.println("Permission Denial: Can't dump DropBoxService");
                                        …
            pw.println("Can't initialize: " + e);
                                        …


                pw.print("Unknown argument: ");
                pw.println(args[i]);
                                        …
        pw.println();
                                        …
            pw.print("Searching for:");
                                        …
            pw.println();
                                        …
        pw.println();
                                        …
            pw.print(date);
            pw.print(" ");
            pw.print(entry.tag == null ? "(no tag)" : entry.tag);
                                        …
                pw.println(" (no file)");
                                        …
                pw.println(" (contents lost)");
                                        …
                pw.print((entry.flags & DropBox.IS_GZIPPED) != 0 ? "
(comopressed " : " (");
                pw.print((entry.flags & DropBox.IS_TEXT) != 0 ? "text" :
"data");
```

```
                                …
            pw.println();
                                …
            pw.println(entry.file.getPath());
                                …
                    pw.print("    ");
                    pw.print(text.trim().replace('\n', '/'));
                                …
                                 pw.print(" ...");
                    pw.println();
                                …
                pw.print("*** ");
                pw.println(e.toString());
                                …
                                 pw.println();
                                …
                            pw.println("(No entries found.)");
                                …
                         pw.println();
        pw.println("Usage: dumpsys dropbox [--print|--file] [YYYY-mm-dd]
[HH:MM:SS.SSS] [tag]");
                                … .
```

See https://android.googlesource.com/platform/frameworks/base/+/897a744bb7a95b2d3883004301b8e877cd5efc92/services/java/com/android/server/DropBoxService.java.

41. Said print() or println() statements of the PrintWriter method contained in Defendant Google's infringing code specifically comprise the following print() or println() statements of the PrintWriter method having an out parameter:

```
out = new PrintWriter(new FileOutputStream(randomDevice));
```

```
            out.println("Copyright (C) 2009 The Android Open Source
Project");

            out.println("All Your Randomness Are Belong To Us");

            out.println(START_TIME);

            out.println(START_NANOTIME);

            out.println(SystemProperties.get("ro.serialno"));

            out.println(SystemProperties.get("ro.bootmode"));

            out.println(SystemProperties.get("ro.baseband"));

            out.println(SystemProperties.get("ro.carrier"));

            out.println(SystemProperties.get("ro.bootloader"));

            out.println(SystemProperties.get("ro.hardware"));

            out.println(SystemProperties.get("ro.revision"));

            out.println(SystemProperties.get("ro.build.fingerprint"));

            out.println(new Object().hashCode());

            out.println(System.currentTimeMillis());

            out.println(System.nanoTime());
                                        … .
```

See https://android.googlesource.com/platform/frameworks/base/+/0e2d281/services/java/com/android/server/EntropyMixer.java.

42. Said print() or println() statements of the PrintWriter method contained in Defendant Google's infringing code specifically comprise the following print() or println() statements of the PrintWriter method having a pw parameter:

```
pw.print("    "); pw.print(SIMPLE_NAME); pw.println(":");

pw.print("      mConnected="); pw.println(mConnected);
```

```
pw.print("      mRegistered="); pw.println(mRegistered);

pw.print("      mBootComplete="); pw.println(mBootComplete);
                                        …

    pw.println("      mSubscriptions=");
                                        …

        pw.print("        ");

        pw.println(conditionId);
                                        …
pw.println("      mTrackers=");
                                        …
    pw.print("        user="); pw.println(mTrackers.keyAt(i));
                                        … .
```

See

https://android.googlesource.com/platform/frameworks/base/+/master/services/core/java/com/android/server/notification/EventConditionProvider.java?autodive=0%2F%2F.

43. Said print() or println() statements of the PrintWriter method contained in Defendant Google's infringing code specifically comprise the following print() or println() statements of the PrintWriter method having a pw parameter:

```
pw.print("PollingIntervalMs: ");
                                        …
pw.print("\nPollingIntervalShorterMs: ");
                                        …
pw.println("\nTryAgainTimesMax: " + mTryAgainTimesMax);

pw.print("TimeErrorThresholdMs: ");
                                        …
pw.println("\nTryAgainCounter: " + mTryAgainCounter);

pw.print("LastNtpFetchTime: ");
                                        …
pw.println();
                                        … .
```

See https://android.googlesource.com/platform/frameworks/base.git/+/master/services/core/java/com/android/server/NetworkTimeUpdateService.java.

44. Said print() or println() statements of the PrintWriter method contained in Defendant Google's infringing code specifically comprise the following print() or println() statements of the PrintWriter method having a pw parameter:

```
pw.println(prefix + this);

            pw.println(prefix + "  icon=0x" +
Integer.toHexString(notification.icon)

                    + " / " + idDebugString(baseContext, this.pkg,
notification.icon));

            pw.println(prefix + "  pri=" + notification.priority);

            pw.println(prefix + "  score=" + this.score);

            pw.println(prefix + "  contentIntent=" +
notification.contentIntent);

            pw.println(prefix + "  deleteIntent=" +
notification.deleteIntent);

            pw.println(prefix + "  tickerText=" + notification.tickerText);

            pw.println(prefix + "  contentView=" + notification.contentView);

            pw.println(prefix + "  uid=" + uid + " userId=" + userId);

            pw.println(prefix + "  defaults=0x" +
Integer.toHexString(notification.defaults));

            pw.println(prefix + "  flags=0x" +
Integer.toHexString(notification.flags));

            pw.println(prefix + "  sound=" + notification.sound);

            pw.println(prefix + "  vibrate=" +
Arrays.toString(notification.vibrate));
```

```
            pw.println(prefix + "  ledARGB=0x" +
Integer.toHexString(notification.ledARGB)

                    + " ledOnMS=" + notification.ledOnMS

                    + " ledOffMS=" + notification.ledOffMS);
                                        …
pw.println(prefix + this);
                                        …
pw.println("Permission Denial: can't dump NotificationManager from from pid="

                    + Binder.getCallingPid()

                    + ", uid=" + Binder.getCallingUid());
                                        …

     pw.println("Current Notification Manager state:");

                                        …

pw.println("  Toast Queue:");

                pw.println("  ");
                                        …

                pw.println("  Notification List:");
                                        …

                pw.println("  ");
                                        …
                pw.println("  Lights List:");

                                        …
                pw.println("  ");
                                        …
            pw.println("  mSoundNotification=" + mSoundNotification);

            pw.println("  mSound=" + mSound);

            pw.println("  mVibrateNotification=" + mVibrateNotification);

            pw.println("  mDisabledNotifications=0x" +
Integer.toHexString(mDisabledNotifications));

            pw.println("  mSystemReady=" + mSystemReady);
```

```
                                        …
pw.println("Permission Denial: can't dump NotificationManager from from pid="
                    + Binder.getCallingPid()
                    + ", uid=" + Binder.getCallingUid());
                                        …
pw.println("Current Notification Manager state:");
                                        …
                pw.println("  Toast Queue:");
                                        …
                pw.println("  ");
                                        …
                pw.println("  Notification List:");
                                        …
                pw.println("  ");
                                        …
                pw.println("  Lights List:");
                                        …
                pw.println("  ");
                                        …
            pw.println("  mSoundNotification=" + mSoundNotification);
            pw.println("  mVibrateNotification=" + mVibrateNotification);
            pw.println("  mDisabledNotifications=0x" +
Integer.toHexString(mDisabledNotifications));
            pw.println("  mSystemReady=" + mSystemReady);
                                        … .
```

See https://android.googlesource.com/platform/frameworks/base.git/+/android-4.2.2_r1/services/java/com/android/server/NotificationManagerService.java.

45. Said print() or println() statements of the PrintWriter method contained in Defendant Google's infringing code specifically comprise the following print() or println() statement of the PrintWriter method having an out parameter:

```
pw.println(dateString + ": " + msg);
                                        …
out.print(prefix); out.print(
```

```
Integer.toHexString(System.identityHashCode(filter.activity)));

                    out.print(' ');

                                        …
out.print(" filter ");


out.println(Integer.toHexString(System.identityHashCode(filter)));

                                        …
out.print(prefix); out.print(
                                        …

out.print(prefix);

            out.print(


Integer.toHexString(System.identityHashCode(filter.provider)));

            out.print(' ');

            filter.provider.printComponentShortName(out);

            out.print(" filter ");


out.println(Integer.toHexString(System.identityHashCode(filter)));
                                        … .
```

See https://android.googlesource.com/platform/frameworks/base/+/a029ea1/services/java/com/android/server/pm/PackageManagerService.java.

46. Said print() or println() statements of the PrintWriter method contained in Defendant Google's infringing code specifically comprise the following print() or println() statements of the PrintWriter method having a pw parameter:

```
pw.print("mLowBatteryAlertCloseLevel=");
```

```
        pw.println(mLowBatteryAlertCloseLevel);

        pw.print("mLowBatteryReminderLevels=");

        pw.println(Arrays.toString(mLowBatteryReminderLevels));

        pw.print("mInvalidChargerDialog=");

        pw.println(mInvalidChargerDialog == null ? "null" :
mInvalidChargerDialog.toString());

        pw.print("mLowBatteryDialog=");

        pw.println(mLowBatteryDialog == null ? "null" :
mLowBatteryDialog.toString());

        pw.print("mBatteryLevel=");

        pw.println(Integer.toString(mBatteryLevel));

        pw.print("mBatteryStatus=");

        pw.println(Integer.toString(mBatteryStatus));

        pw.print("mPlugType=");

        pw.println(Integer.toString(mPlugType));

        pw.print("mInvalidCharger=");

        pw.println(Integer.toString(mInvalidCharger));

        pw.print("bucket: ");

        pw.println(Integer.toString(findBatteryLevelBucket(mBatteryLevel)));
```

See

https://android.googlesource.com/platform/frameworks/base/+/6b25e72/packages/SystemUI/src/com/android/systemui/power/PowerUI.java.

47. Said print() or println() statements of the PrintWriter method contained in Defendant Google's infringing code specifically comprise the following print() or println() statements of the PrintWriter method having a pw parameter:

```
pw.print("    "); pw.print(SIMPLE_NAME); pw.println(":");

        pw.print("      mConnected="); pw.println(mConnected);

        pw.print("      mRegistered="); pw.println(mRegistered);

        pw.println("      mSubscriptions=");
                                        …
                pw.print("        ");

                pw.print(meetsSchedule(mSubscriptions.get(conditionId), now)
? "* " : "  ");

                pw.println(conditionId);

                pw.print("            ");

                pw.println(mSubscriptions.get(conditionId).toString());
                                        …
        pw.println("      snoozed due to alarm: " + TextUtils.join(SEPARATOR,
mSnoozed));
                                      … .
```

See

https://android.googlesource.com/platform/frameworks/base/+/master/services/core/java/com/android/server/notification/ScheduleConditionProvider.java.

48. Said print() or println() statements of the PrintWriter method contained in Defendant Google's infringing code specifically comprise the following print() or println() statements of the PrintWriter method having a pw parameter:

```
pw.println("\n  Snoozed notifications:");
                                        …
            pw.print(INDENT);

            pw.println("user: " + userId);
                                        …
                pw.print(INDENT);
```

```
                pw.print(INDENT);

                pw.println("package: " + pkg);
                                        …
                    pw.print(INDENT);

                    pw.print(INDENT);

                    pw.print(INDENT);

                    pw.println(key);
                                        … .
```

See https://android.googlesource.com/platform/frameworks/base/+/master/services/core/java/com/android/server/notification/SnoozeHelper.java?autodive=0%2F%2F.

49. Said print() or println() statements of the PrintWriter method contained in Defendant Google's infringing code specifically comprise the following print() or println() statements of the PrintWriter method having a pw parameter:

```
pw.println("Current UI Mode Service state:");

            pw.print("  mDockState="); pw.print(mDockState);

                    pw.print(" mLastBroadcastState=");
pw.println(mLastBroadcastState);

            pw.print("  mNightMode="); pw.print(mNightMode);

                    pw.print(" mNightModeLocked=");
pw.print(mNightModeLocked);

                    pw.print(" mCarModeEnabled="); pw.print(mCarModeEnabled);

                    pw.print(" mComputedNightMode=");
pw.print(mComputedNightMode);

                    pw.print(" mCarModeEnableFlags=");
pw.print(mCarModeEnableFlags);
```

```
                    pw.print(" mEnableCarDockLaunch=");
pw.println(mEnableCarDockLaunch);

            pw.print("  mCurUiMode=0x");
pw.print(Integer.toHexString(mCurUiMode));

                    pw.print(" mUiModeLocked="); pw.print(mUiModeLocked);

                    pw.print(" mSetUiMode=0x");
pw.println(Integer.toHexString(mSetUiMode));

            pw.print("  mHoldingConfiguration=");
pw.print(mHoldingConfiguration);

                    pw.print(" mSystemReady="); pw.println(mSystemReady);

            if (mTwilightManager != null) {

                // We may not have a TwilightManager.

                pw.print("  mTwilightService.getLastTwilightState()=");

                pw.println(mTwilightManager.getLastTwilightState());
                                        …

pw.println("UiModeManager service (uimode) commands:");

            pw.println("  help");

            pw.println("    Print this help text.");

            pw.println("  night [yes|no|auto]");

            pw.println("    Set or read night mode.");
                                        …

    pw.println("Night mode: " + currModeStr);

                                        … .
```

See

https://android.googlesource.com/platform/frameworks/base/+/master/services/core/java/com/android/server/UiModeManagerService.java.

50. Said print() or println() statements of the PrintWriter method contained in Defendant Google's infringing code specifically comprise the following print() or println() statements of the PrintWriter method having a pw parameter:

```
pw.println("USB Manager State:");
                                        …
                pw.println("Settings for user " + userId + ":");
                                        … .
```

See https://android.googlesource.com/platform/frameworks/base/+/e098050/services/java/com/android/server/usb/UsbService.java.

51. Said print() or println() statements of the PrintWriter method contained in Defendant Google's infringing code specifically comprise the following print() or println() statements of the PrintWriter method having a pw parameter:

```
pw.print("WatchdogStatus: ");

pw.print("State " + getCurrentState());

pw.println(", network [" + mConnectionInfo + "]");

pw.print("checkFailures    " + mNumCheckFailures);

pw.println(", bssids: " + mBssids);

pw.println("lastSingleCheck: " + mOnlineWatchState.lastCheckTime);
                                        … .
```

See https://android.googlesource.com/platform/frameworks/base.git/+/ics-mr0-release/wifi/java/android/net/wifi/WifiWatchdogStateMachine.java.

COUNT I: DEFENDANT GOOGLE INFRINGEMENT

OF U.S. PATENT 8,713,425

52. Progme re-alleges paragraphs 1-51 as if fully set forth herein.

53. At least since Defendant Google received service of this Complaint, Defendant Google has had knowledge of the ‘425 Patent-in-suit or has been willfully blind to the existence of the ‘425 Patent-in-suit.

54. On information and belief, Defendant Google continues to infringe the ‘425 Patent after being made aware of the existence and Defendant Google’s infringement of the ‘425 Patent at least from actual notice from service of this Complaint.

55. Defendant Google has been and is now directly infringing and/or indirectly infringing the ‘425 Patent by way of inducement and/or contributory infringement, literally and/or under the doctrine of equivalents, in this District, and elsewhere, in violation of 35 U.S.C. § 271, including by making, using, selling and/or offering for sale in the United States or importing into the United States, one or more Android SDKs and/or Android applications generating and/or encoding one or more print() or println() statements of the PrintWriter method listed in **Defendant Google’s Infringing Code** documented above. Defendant Google contributes to and induces the direct infringement of the ‘425 Patent by both Android application developers and Android device manufacturers including the distribution of Android SDK and specifically including **Defendant Google’s Infringing Code** documented above. Defendant Google contributes to and induces the direct infringement of the ‘425 Patent by Android application developers and specifically including incorporating Android operating systems and Android applications in Android devices generating and/or encoding one or more print() or println() statements of the PrintWriter method listed in **Defendant Google’s Infringing Code** documented above.

56. Defendant Google actively and knowingly has infringed and infringes the ‘425 Patent with knowledge of Progme’s patent rights and without reasonable basis for believing that

Defendant Google's conduct is lawful. Defendant Google has also induced and contributed and continues to induce and contribute to the infringement of the '425 Patent by third party developers developing applications using one or more portions of **Defendant Google's Infringing Code** documented above. Defendant Google's acts of infringement have been and continue to be willful, deliberate and in reckless disregard of, or alternatively willfully blind to, Progme's patent rights. Defendant Google is thus liable to Progme for infringement of the '425 Patent pursuant to 35 USC § 271.

57. Defendant Google in its BroadcastReceiver Component system, by generating and/or encoding one or more portions of **Defendant Google's Infringing Code** documented above, has been and is now infringing the '425 Patent, directly and indirectly by way of inducement and/or contributory infringement, by using "Request and Response parameters included in the one or more predetermined parameters defining the pre-defined printable output of the predetermined hyperlinked content indicated in the second attribute in the hyperlink address string [to] … encapsulate the data sent by the client." *See* the '425 Patent at 14:62-67 – 15:1.

58. Defendant Google in its BroadcastReceiver Component system, by generating and/or encoding one or more portions of **Defendant Google's Infringing Code** documented above, has been and is now infringing the '425 Patent, directly and indirectly by way of inducement and/or contributory infringement, by using hyperlinking and PrintWriter printing to generate dynamically generated information comprising "configuration data at initialization time … [allowing] different instances of the same … class to be initialized with different data, and be managed as differently named … [wherein the] data provided

at initialization time includes an area where each instance keeps its persistent instance-specific state ". *See* the '425 Patent at 15:15 - 20.

59. Defendant Google in its BroadcastReceiver Component system, by generating and/or encoding one or more portions of **Defendant Google's Infringing Code** documented above, has been and is now infringing the '425 Patent, directly and indirectly by way of inducement and/or contributory infringement, by using hyperlinking and PrintWriter printing to generate dynamically generated information comprising "arguments to an instantiated … object through the [Java programming language] properties object" as "a set of 'name:value' pairs … ". *See* the '425 Patent at pg. 17, cols. 58-67 – pg. 18, cols. 1-3.

60. In addition to its direct infringement, Defendant Google has been and is now indirectly infringing by way of inducing infringement and/or contributing to the infringement of one or more claims of the '425 Patent.

61. Defendant Google indirectly infringes at least one claim of the '425 Patent.

62. At least since obtaining knowledge of the '425 Patent, Defendant Google has indirectly infringed and continues to indirectly infringe the '425 Patent by actively inducing infringement of one or more method claims 2-13 of the '425 Patent by third party developers in violation of 35 USC § 271(b) and contributorily infringing one or more of method claims 2-13 of the '425 Patent based on third party developers' direct infringement in violation of 35 USC § 271(c).

63. Defendant Google directly infringes at least claims 2 and 4 as well as claims 5 or 7 of the '425 Patent by generating one or more print() or println() statements of the PrintWriter method listed in **Defendant Google's Infringing Code** documented above as "source

tree" code stored online at android.com at Git hosted by Defendant Google wherein a resource in the initial array position of a list of resource identifiers uniquely identifying resources is defined within a print() or println() statement of the PrintWriter method, a parameter instructing a PrintWriter to print predetermined printable output of said resource in the initial array position defined within said print() or println() statement of the PrintWriter method comprises an out or pw parameter and said predetermined printable output of said resource in the initial array position defined within said print() or println() statement of the PrintWriter method is printed via said PrintWriter.

64. Under claims 2 and 4 as well as claims 5 or 7 of the '425 Patent, Defendant Google first generates as an hyperlink address string structured as a PrintWriter method one or more print() or println() statements of the PrintWriter method listed in **Defendant Google's Infringing Code** documented above and then actively promotes and encourages third party developers to generate said hyperlink address string structured as a PrintWriter method by downloading one or more print() or println() statements of the PrintWriter method listed in **Defendant Google's Infringing Code** documented above.

65. Defendant Google first generates one or more print() or println() statements of the PrintWriter method listed in **Defendant Google's Infringing Code** documented above when said one or more statements are stored at the respective android.com web addresses documented above.

66. Defendant Google's generating one or more print() or println() statements of the PrintWriter method listed in **Defendant Google's Infringing Code** documented above constitutes direct infringement of the '425 Patent by Defendant Google.

67. Defendant Google generates one or more print() or println() statements of the PrintWriter method listed in **Defendant Google's Infringing Code** documented above to be transmitted in conjunction with program signals representative of predetermined program material by Defendant Google's BroadcastReceiver Component system.

68. By generating one or more print() or println() statements of the PrintWriter method listed in **Defendant Google's Infringing Code** documented above to be transmitted in conjunction with program signals representative of predetermined program material by Defendant Google's BroadcastReceiver Component system, Defendant Google directs and controls certain program signal generating and/or encoding activities of third party developers that directly infringe the '425 Patent.

69. On information and belief, since receiving service of this Complaint, Defendant Google has generated and continues to generate one or more print() or println() statements of the PrintWriter method listed in **Defendant Google's Infringing Code** documented above to be transmitted in conjunction with program signals representative of predetermined program material via Defendant Google's BroadcastReceiver Component system.

70. Third party developers, immediately upon download of one or more print() or println() statements of the PrintWriter method listed in **Defendant Google's Infringing Code** documented above, directly infringe the '425 Patent by generating, by said download, said print() or println() statements of the PrintWriter method to be transmitted in conjunction with program signals representative of predetermined program material via Defendant Google's BroadcastReceiver Component system.

71. Third party developers, by generating one or more print() or println() statements of the PrintWriter method listed in **Defendant Google's Infringing Code** documented above to

be transmitted in conjunction with program signals representative of predetermined program material via Defendant Google's BroadcastReceiver Component system, directly infringe the '425 Patent.

72. Defendant Google encodes one or more print() or println() statements of the PrintWriter method listed in **Defendant Google's Infringing Code** documented above to be transmitted in conjunction with program signals representative of predetermined program material via Defendant Google's BroadcastReceiver Component system.

73. On information and belief, a third party developer, each time before generating one or more print() or println() statements of the PrintWriter method listed in **Defendant Google's Infringing Code** documented above, negotiated and continues to negotiate an agreement with Defendant Google that said one or more print() or println() statements of the PrintWriter method were to and will be generated by third party developers for Defendant Google to encode and transmit said one or more print() or println() statements of the PrintWriter method in conjunction with program signals representative of predetermined program material via Defendant Google's BroadcastReceiver Component system.

74. On information and belief, said agreement that one or more print() or println() statements of the PrintWriter method listed in **Defendant Google's Infringing Code** documented above were to and will be generated by third party developers for Defendant Google to encode and transmit said one or more print() or println() statements of the PrintWriter method in conjunction with program signals representative of predetermined program material via Defendant Google's BroadcastReceiver Component system induced and

continues to induce third party developers to generate said print() or println() statements of the PrintWriter method.

75. In other words, on information and belief, in exchange for agreeing to encode and transmit one or more print() or println() statements of the PrintWriter method listed in **Defendant Google's Infringing Code** documented above in conjunction with program signals representative of predetermined program material via Defendant Google's BroadcastReceiver Component system, third party developer agreed and continues to agree to generate said one or more print() or println() statements of the PrintWriter method.

76. On information and belief, Defendant Google was aware or was willfully blind to the fact that its agreeing to encode and transmit said one or more print() or println() statements of the PrintWriter method listed in **Defendant Google's Infringing Code** documented above generated by third party developer in conjunction with program signals representative of predetermined program material via Defendant Google's BroadcastReceiver Component system would induce third party developers to generate said one or more print() or println() statements of the PrintWriter method.

77. Defendant Google's active inducement of infringement and contributory infringement has occurred with actual knowledge of the '425 Patent.

78. Defendant Google's active inducement of infringement has occurred with the specific intent of encouraging others to infringe or willful blindness to the fact that its activities would induce infringement of the '425 Patent as demonstrated by, *inter alia*, providing specifications and instructions for the installation and operation of one or more print() or println() statements of the PrintWriter method listed in **Defendant Google's Infringing**

Code documented above, including uses that infringe one or more claims of the '425 Patent and/or promote and encourage and/or aiding others through contracts, agreements and/or computerized instructions performance of one or more of the methods claimed in claims 2-13 of the '425 Patent.

79. On information and belief, Defendant Google has purposefully, actively and voluntarily distributed or promoted the use of said one or more print() or println() statements of the PrintWriter method listed in **Defendant Google's Infringing Code** documented above contained in the Android BroadcastReceiver Component system for applications deploying said infringing code with the expectation that said applications will be purchased, used and/or licensed by predetermined application developers, *inter alia*, in the Eastern District of Michigan. Defendant Google has thereby committed acts of infringement of the '425 Patent in this judicial district. By purposefully and voluntarily distributing or promoting the use of said infringing code, Defendant Google has injured Progme and is thus liable to Progme for infringement of the '425 Patent pursuant to 35 USC § 271.

80. Defendant Google actively promotes and encourages third party developers to download one or more print() or println() statements of the PrintWriter method listed in **Defendant Google's Infringing Code** documented above from "Android source tree … located in a Git repository hosted by Google". *See* https://source.android.com/setup/build/downloading.

81. By actively promoting and encouraging said third party developers to download one or more print() or println() statements of the PrintWriter method listed in **Defendant Google's Infringing Code** documented above, Defendant Google promotes and

encourages said third party developers to generate one or more print() or println() statements of the PrintWriter method listed in **Defendant Google's Infringing Code** documented above and thereby directly infringe at least claims 2 and 4 as well as claims 5 or 7 of the '425 Patent.

82. By downloading one or more print() or println() statements of the PrintWriter method listed in **Defendant Google's Infringing Code** documented above to third party developers, Defendant Google directs and controls the direct infringement by said third party developers in generating said infringing source code, thereby making Defendant Google vicariously liable for said third party developer's direct infringement.

83. On information and belief, Defendant Google induces and continues to induce infringement of the '425 Patent with specific intent that the induced acts of third party developers downloading one or more print() or println() statements of the PrintWriter method listed in **Defendant Google's Infringing Code** documented above constitute infringement of the '425 Patent.

84. One information and belief, Defendant Google's contributory infringement has occurred with knowledge that one or more print() or println() statements of the PrintWriter method listed in **Defendant Google's Infringing Code** documented above is or are a material part of the invention and knowing the same to be especially made or especially adopted for use in an infringement of the '425 Patent and not a staple article or commodity of commerce suitable to substantial non-infringing use. For example, said one or more print() or println() statements of the PrintWriter method listed in **Defendant Google's Infringing Code** documented above is or are specifically designed for use in infringement of the '425 Patent. Due to the specific design of said one or more statements as the claimed hyperlink

address string structured as a PrintWriter method, said one or more print() or println() statements of the PrintWriter method listed in **Defendant Google's Infringing Code** documented above do not have any substantial non-infringing use.

COUNT II: DEFENDANT GOOGLE INFRINGEMENT

OF U.S. PATENT 8,713,425

85. Progme re-alleges paragraphs 1-51 as if fully set forth herein.

86. At least since Defendant Google received service of this complaint, Defendant Google has had knowledge of the '425 Patent-in-suit or has been willfully blind to the existence of the '425 Patent-in-suit.

87. Defendant Google continued to infringe the '425 Patent after being made aware of the existence of the '425 Patent from actual notice from said service of this complaint.

88. On information and belief, Defendant Google has directly infringed one or more of claims 14-25 of the '425 Patent by making, using (including using for testing, debugging and diagnosis purposes), importing, offering for sale and/or selling within the United States the Accused Instrumentalities specified below in violation of 35 U.S.C. § 271(a).

89. On information and belief, said Accused Instrumentalities comprise an apparatus as disclosed and claimed in the '425 Patent for receiving an hyperlink address string structured as a PrintWriter method in conjunction with predetermined program material and processing A) a predetermined hyperlink address comprising said resource identifier to hyperlink to said resource in the initial array position of said list in which resource identifiers uniquely identifying resources corresponding to predetermined program material are arrayed and B) a parameter instructing a PrintWriter to print predetermined printable output of said resource in the initial array position indicated in a second attribute

of said hyperlink address string. See the '425 Patent, for example, at pg. 3, cols. 33-41, pg. 21, cols. 6-10 and claim 14.

90. Said Accused Instrumentalities sold, leased or otherwise used in direct infringement of the '425 Patent by Defendant Google comprise Android Runtime BroadcastReceiver devices.

91. Said Accused Instrumentalities deploy a java virtual machine enabling said Android Runtime to perform the receiving and processing functions claimed in the '425 Patent.

92. Said Accused Instrumentalities include the receiving apparatus disclosed and claimed in the '425 Patent comprising virtual machine functionality designed to receive and process as claimed therein said print() or println() statements of the PrintWriter method listed in **Defendant Google's Infringing Code** documented above.

RELIEF WARRANTED FOR INFRINGEMENT OF U.S. PATENT 8,713,425

93. Defendant Google's infringing activity alleged above comprises the compelling reason Defendant Google's BroadcastReceiver Component system and said Android Runtime is used by third party developers and acquired in the consumer marketplace.

94. Defendant Google's infringing activity alleged above creates a performance advantage in Defendant Google's BroadcastReceiver Component system and said Android Runtime that drives demand for Defendant Google's BroadcastReceiver Component system and said Android Runtime.

95. Progme has no adequate remedy at law against Defendant Google's acts of infringement and, unless Defendant Google is enjoined from continuing to infringe the '425 Patent, Progme will suffer irreparable harm. Indeed, the hardships that would be imposed upon Defendant Google by an injunction are relatively less than those faced by Plaintiff Progme

should an injunction not issue. Finally, the public interest would be served by issuance of an injunction.

96. Defendant Google's has had prior constructive notice by marking of the '425 Patent as indicated in **Exhibit C**, the patent number "**Patent 8,713,425**" marked on the PrintHD.TV home page (located at www.printhd.tv) web page, labelling at the bottom of the page on 5/27/14, to provide constructive notice thereof pursuant to 35 U.S.C. § 287.

97. Progme has at all times complied with 35 U.S.C. § 287, providing Defendant Google with prior constructive notice, which constituted consistent and continuous notice of the '425 Patent being infringed to Defendant Google..

98. The method of generating and encoding the hyperlink address string structured as a PrintWriter method claimed in the '425 Patent and alleged infringed herein is capable of being produced in a physical device, a web page, and has been and is noticed in said web page for constructive notice by marking pursuant to 35 U.S.C. § 287.

99. Defendant Google received actual notice of the '425 Patent and Defendant Google's infringement thereof from service of this Complaint.

100. As a result of Defendant Google's acts of infringement, Progme has suffered and will continue to suffer damages in an amount to be proved at trial. Pursuant to 35 U.S.C § 284, Progme is entitled to adequate damages to compensate for infringement including a reasonable royalty from the date of Defendant Google's notice of the '425 Patent. Progme has no means of ascertaining the full extent of Defendant Googles's infringement of the '425 Patent and the amount of Progme's damages resulting from said infringement except through the production of evidence thereof in Defendant Google's sole possession and control.

PRAYER FOR RELIEF

101. WHEREFORE, Progme prays for the following relief:

102. A judgment in favor of Progme that

a. Defendant Google has infringed, directly, literally and/or under the doctrine of equivalents, and indirectly at least one claim of the ’425 Patent;

b. a permanent injunction enjoining Defendant Google and its officers, directors, agents, servants, employees, affiliates, divisions, branches, subsidiaries, parents, and all others acting in concert or privity with any of them from continuing to infringe the ‘425 Patent including from continuing to make, use, sell and/or offer for sale in the United States or import into the United States the Android SDK, Android applications and Android devices and specifically from generating and/or encoding one or more print() or println() statements of the PrintWriter method listed in **Defendant Google’s Infringing Code** documented above;

c. award to Progme the damages to which it is entitled by law and under 35 U.S.C. § 284 for Defendant Google’s past infringement and any continuing or future infringement up until the date Defendant Google is finally and permanently enjoined from further infringement, including both compensatory damages and treble damages for willful infringement;

d. a finding that this is an “exceptional action” and a judgment and order requiring Defendant Google to pay the costs of this action (including all disbursements) as well as attorneys’ fees as provided by 35 U.S.C. § 285;

e. award to Progme pre-judgment and post-judgment interest on its damages and

f. such other further relief in law or equity to which Progme may be justly entitled.

DEMAND FOR JURY TRIAL

103. Pursuant to Rule 38 of the Federal Rules of Civil Procedure, Progme hereby demands a trial by jury as to all issues so triable.

Date: June 1, 2018

Respectfully submitted,

/s/ David A. Reams
David A. Reams, P62855
Law Office of David A. Reams, P.C.
208 Clair Hill Drive
Rochester Hills, MI 48309
248-376-2840
reamslaw@gmail.com